XA-Core On/Off-Switch Algorithm Adjustment

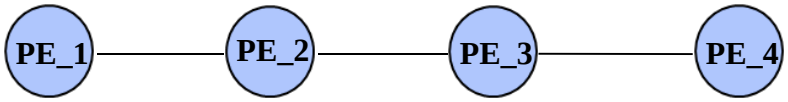
Nicolae Santean – 0S04

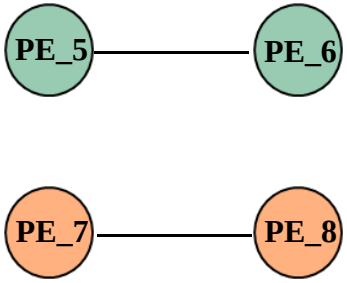
February 2001

Contents

- Defining the problem
- Terms and definitions
- The challenges
- The methodology
- Formulas and the adjustment
- Statistics, the analyzing tool
- Proposal, the on-switch algorithm

Defining the problem - example





BEFORE (the partial de-serialization)

Wb = 10 waitings

AFTER

Wa = 3 waitings

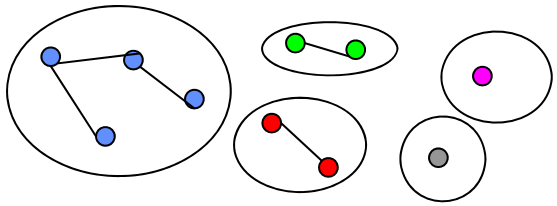
Def. Informally, a **cluster** is a maximal group of processors in contention with each other such that the resulting collision handlers are serialized despite the partial de-serialization.

Terms and definitions

- **Singleton** – a processor not involved in any contention (at a certain given time).
- **Pair** – two processors involved in only one contention: with each other.
- **Cluster** – a maximal group of processors that:
  1. are in contention with each other and
  2. can not be split in two subgroups such that the processors from one subgroup are not in contention with the processors of the other subgroup.
- **Cluster size** – the number of processors of that cluster.

- **Edge probability** – the probability that 2 processes collide: **q** , in the on-switch algorithm.
- **Random graph** – a graph where the edges are taken with a certain probability (edge probability).

Obs. Considering the **contention graph** of the system at a given time, the clusters correspond to the connected components of that graph.



The challenges

- Hard to derive **the number of handlers** generated by a collision cluster .

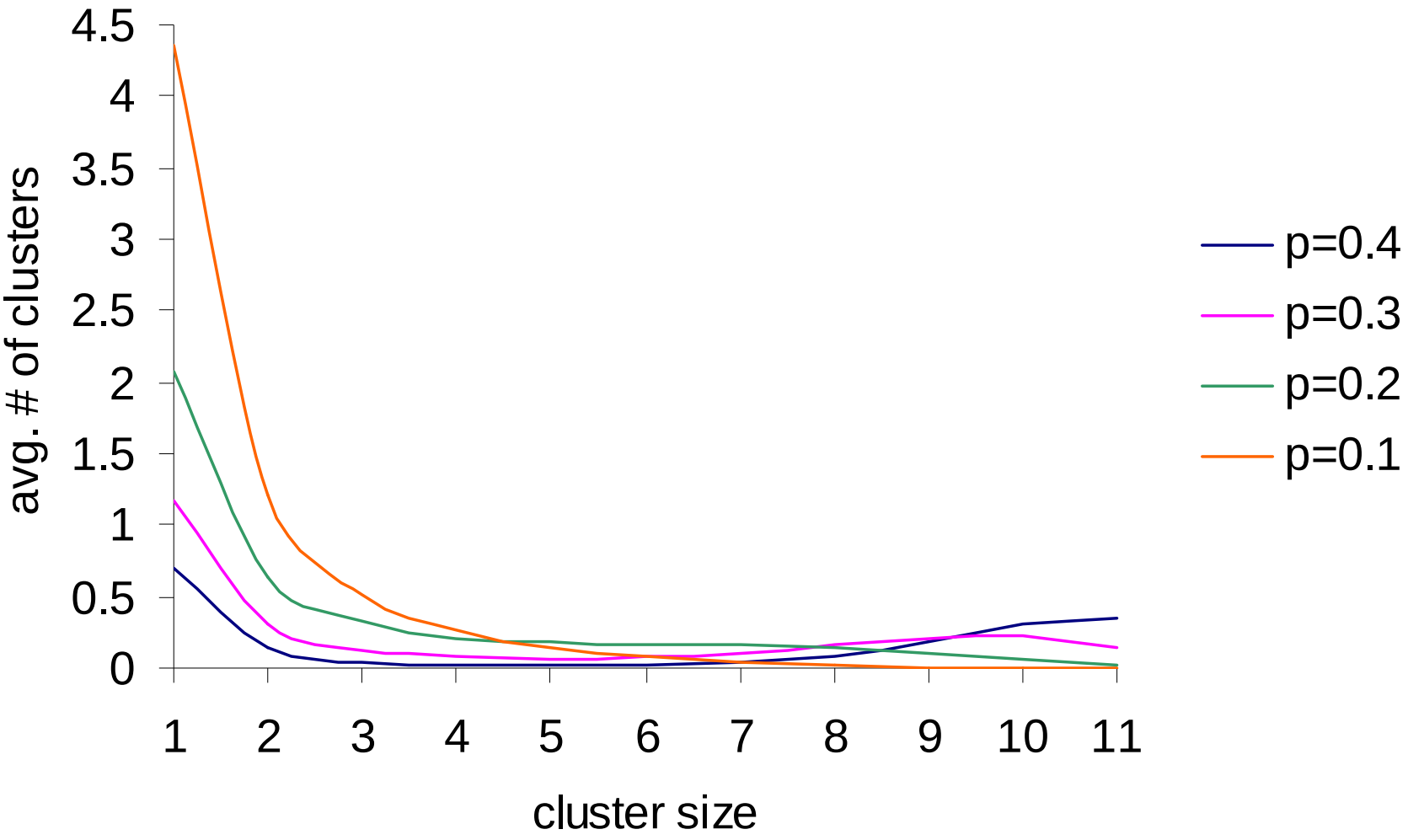
- Hard to describe the **clustering topology** – needs costly measurements; for example measurements for the total number of clusters in the system or for the number of clusters of a certain size etcetera.

- To estimate the **edge probability** (probability that two processes collide) – needs extra measurements. (applicable only for the on-switch algorithm)

- The analytical models are facing the random graph irregularity, for example:
  - **the transition phase**
  - **the giant component**

Obs. **The trivial cases**: (A) when the number of processors is less than 4, or (B) when the total number of clusters is either very small or very large, **lead to almost no difference** in waiting on serialization.

The challenges - the giant component



The methodology

1. **Separate (***the power spent on waiting caused by***) the serialized scheduler by the serialized collision/rollback handler (***at capacity***).**

2. **Consider the worst case scenario, when a cluster of size m generates m-1 serialized handlers (***low impact assumption***)**.

3. **Statistically find the average waiting on serialized collision handlers in both cases: before the change (***partial de-serialization***) and after the change: w_b and w_a .**

4. **Eliminate (average out) the edge-probability** (am I allowed to do this ?)**.**

5. **Compute the scaling factor w_a/w_b .**

6. **Apply (***hardcode***) the scaling to the existing formula.**

Formulas and the adjustment

k = the number of processors.

α_i = the number of clusters of size i.

w_a = the waiting after the change.

w_b = the waiting before the change.

waiting in a cluster of size i

$$w_a = \frac{1}{2} \cdot \sum_{i=1}^{k} \left[\alpha_i \cdot (i-1) \cdot (i-2) \right]$$

$$w_b = \frac{1}{2} \cdot \left(k - \sum_{i=1}^{k} \alpha_i \right) \cdot \left(k - \sum_{i=1}^{k} \alpha_i - 1 \right)$$

total number of handlers in the system

Formulas and the adjustment

k = the number of processors.

s_{rs} = the average serialized part of the collision/rollback handler.

s_s = the average successful transaction time.

B_u = the data contention probability projected to the level expected at the 100% occupancy level of the Rated Power.

x_s = the estimated share of the Useful Processing Power used by the scheduler at capacity.

collision handler share ***scheduler share***

$$w_{s\_old} = \frac{1}{2} \cdot \left[k \cdot \left(s_{rs} \cdot \frac{B_u}{s_s \cdot (1 - B_u)} + x_s \right) \right]^2$$

$$w_{s\_new} = \frac{1}{2} \cdot \left(k \cdot \frac{s_{rs} \cdot B_u}{s_s \cdot (1 - B_u)} \right)^2 \cdot \frac{\overline{w}_a}{\overline{w}_b} + \frac{1}{2} \cdot (k \cdot x_s)^2$$

Statistics - the analyzing tool

D:\XA-CORE>rndgr.exe 7 0.05 100000000

Random Graphs Analyzer v1.a.2.2001

started at Sat Feb 17 00:48:25 2001

{k= 7, p=0.05000, iter=100000000}

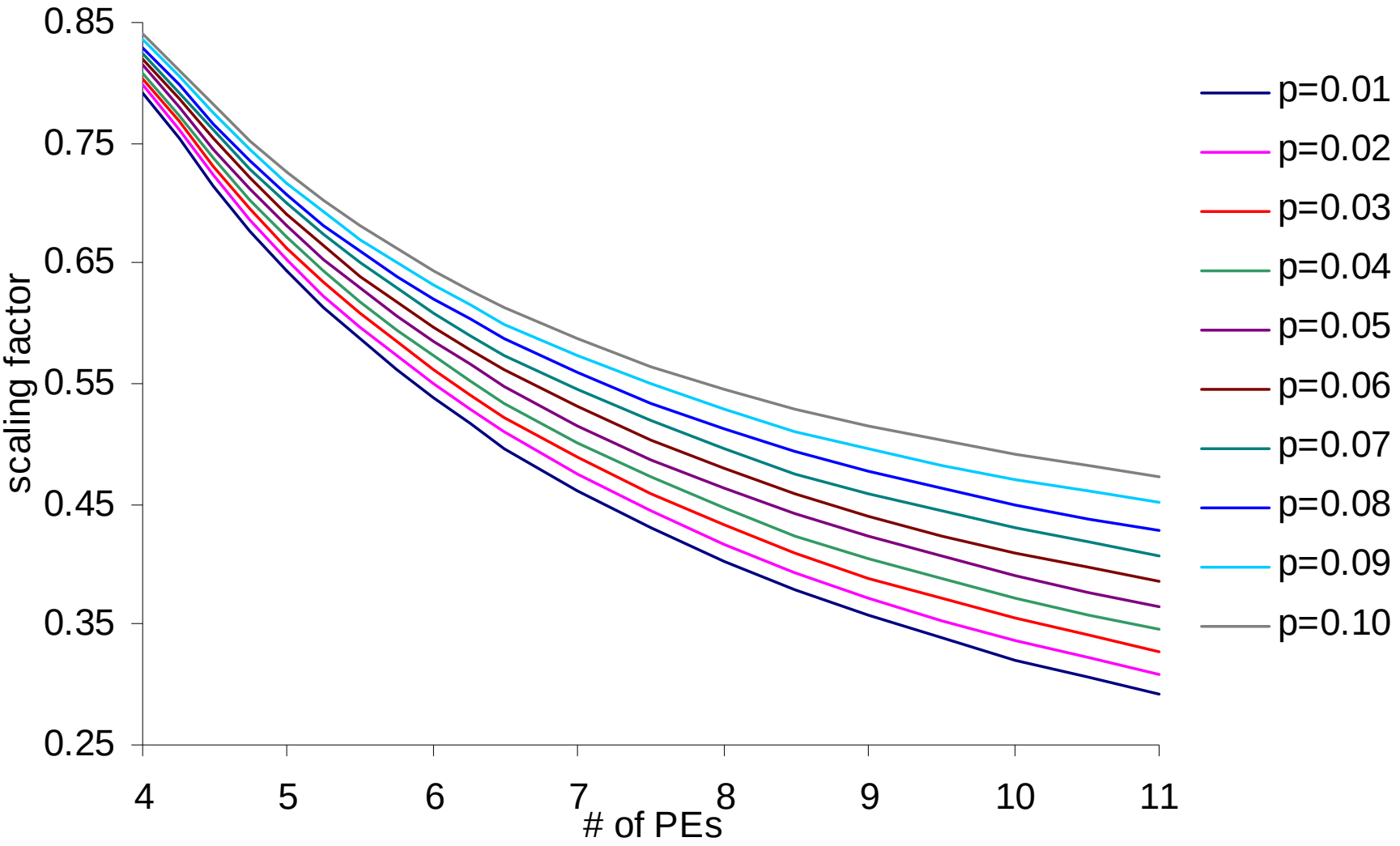
{wa=0.21926, wb=0.42494, scale=0.51598}

[a1:5.18569 a2:0.66581 a3:0.12297 a4:0.02312 a5:0.00371 a6:0.00043 a7:0.00003]

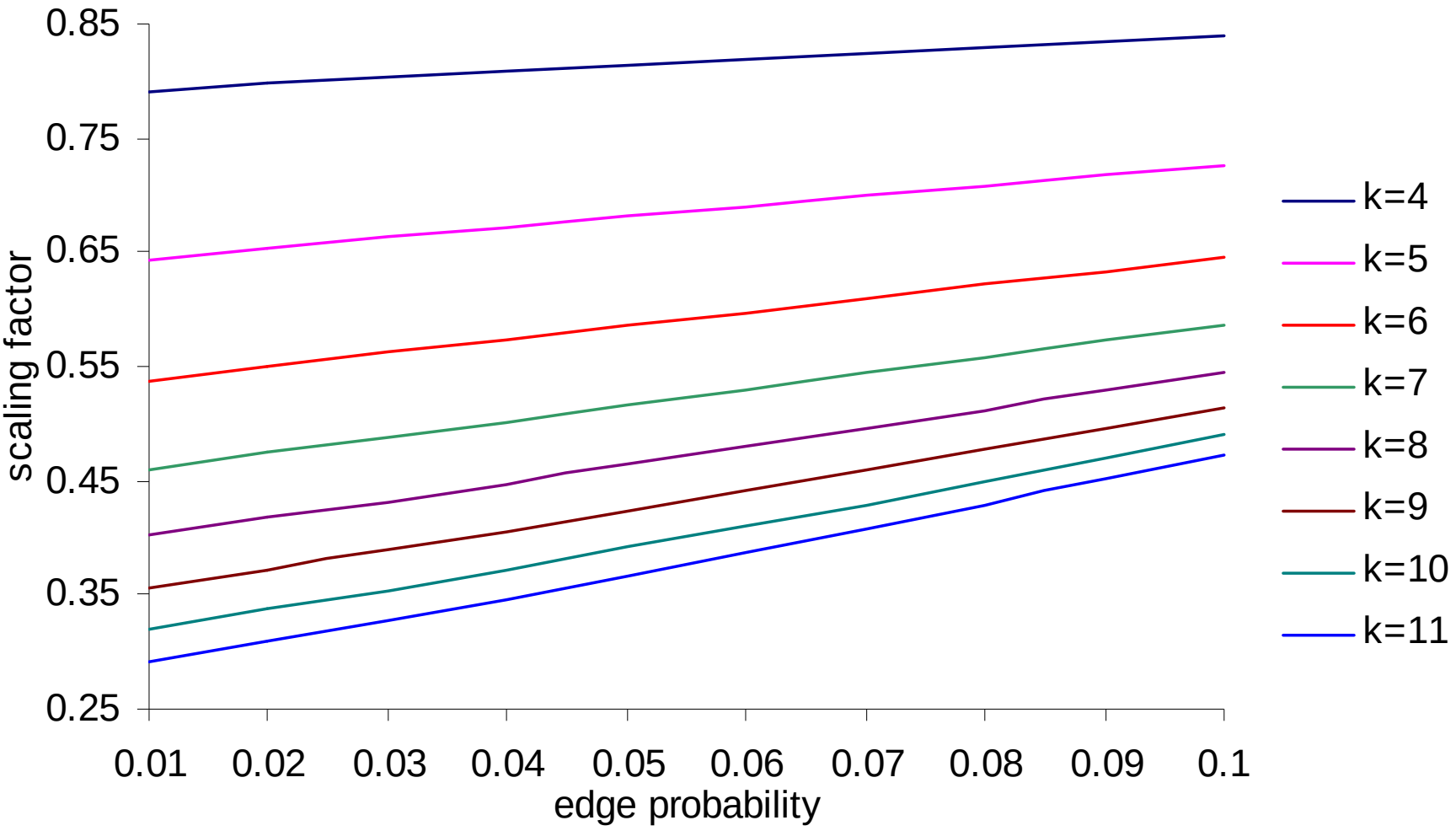
ended at Sat Feb 17 01:06:33 2001

D:\XA-CORE>

Scaling - function of the number of PEs

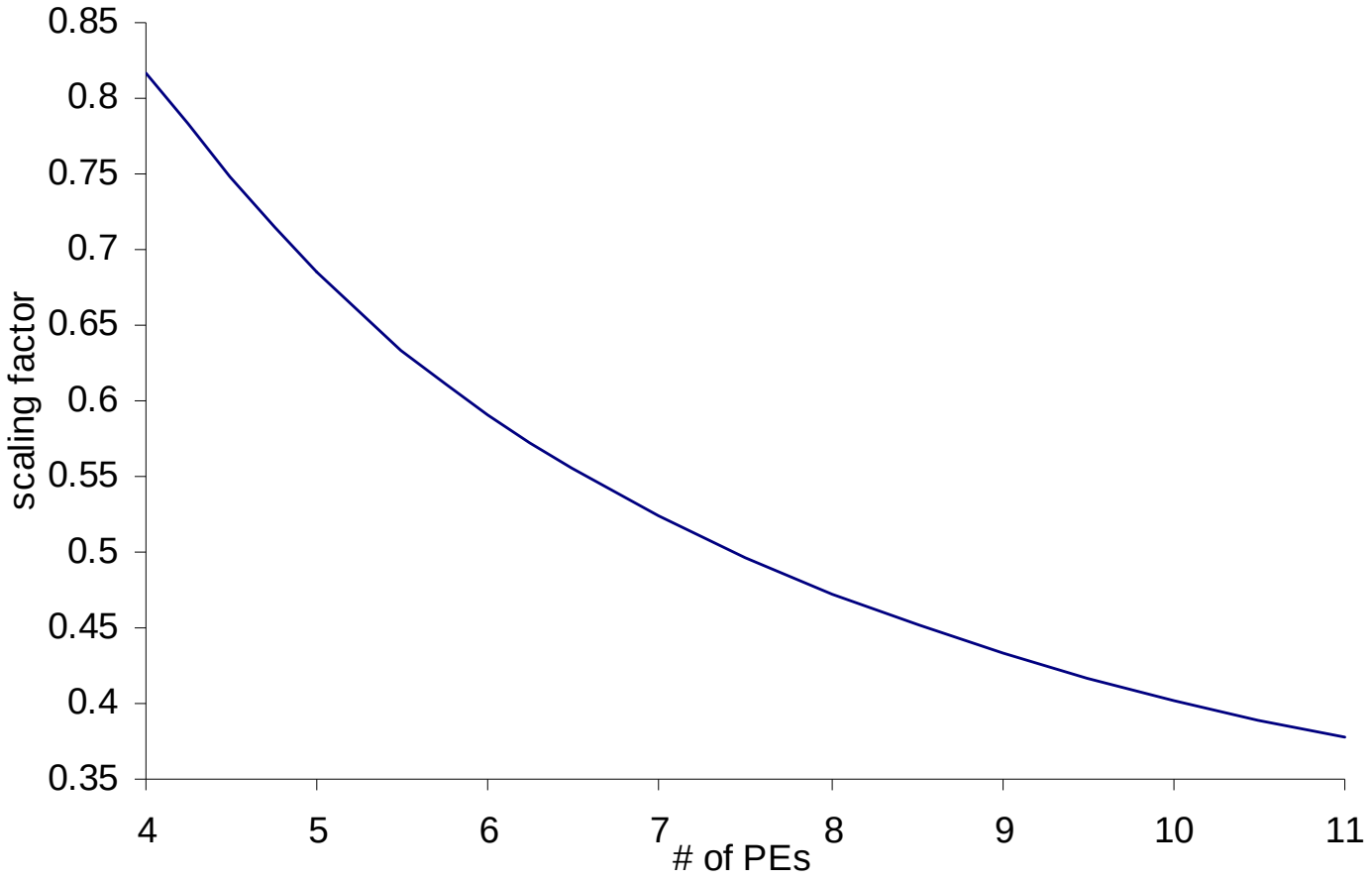


Scaling - function of the edge probability



Average scaling

PEs :	4	5	6	7	8	9	10	11
scale :	0.8166	0.6852	0.5916	0.5233	0.4723	0.4332	0.4026	0.3782



The on-switch algorithm

Step 6 of the algorithm - that computes the processing power spent on waiting by serialized collision/rollback handler and the scheduler at capacity - is adjusted to:

$$w_{s\_new} = \frac{1}{2} \cdot \left(k \cdot \frac{s_{rs} \cdot B_u}{s_s \cdot (1 - B_u)} \right)^2 \cdot \frac{\overline{w}_a}{\overline{w}_b} + \frac{1}{2} \cdot (k \cdot x_s)^2$$

where the scaling factor $\frac{\overline{w}_a}{\overline{w}_b}$ is given by the following table:

PEs :	4	5	6	7	8	9	10	11
scaling :	0.8166	0.6852	0.5916	0.5233	0.4723	0.4332	0.4026	0.3782

Off-switch algorithm adjustment

Choices

EITHER

Use the same methodology as the on-switch adjustment.

OR

Take advantage of the available edge probability **q** and determine the exact scaling.

This can be done by calling the analyzing tool:

rndgr(k, q, 1000000);

instead of averaging out the probability. For k=11 and 1,000,000 iterations, the call takes about 30 seconds for a 16 bit executable on a Pentium III processor at 500MHz.